

Assignment 1

Student Name: Abdul Kalam

Branch: BE-CSE

Semester: 6th

Subject Name: Advanced Programming

UID: 22BCS14739

Section/Group: IOT-601-B

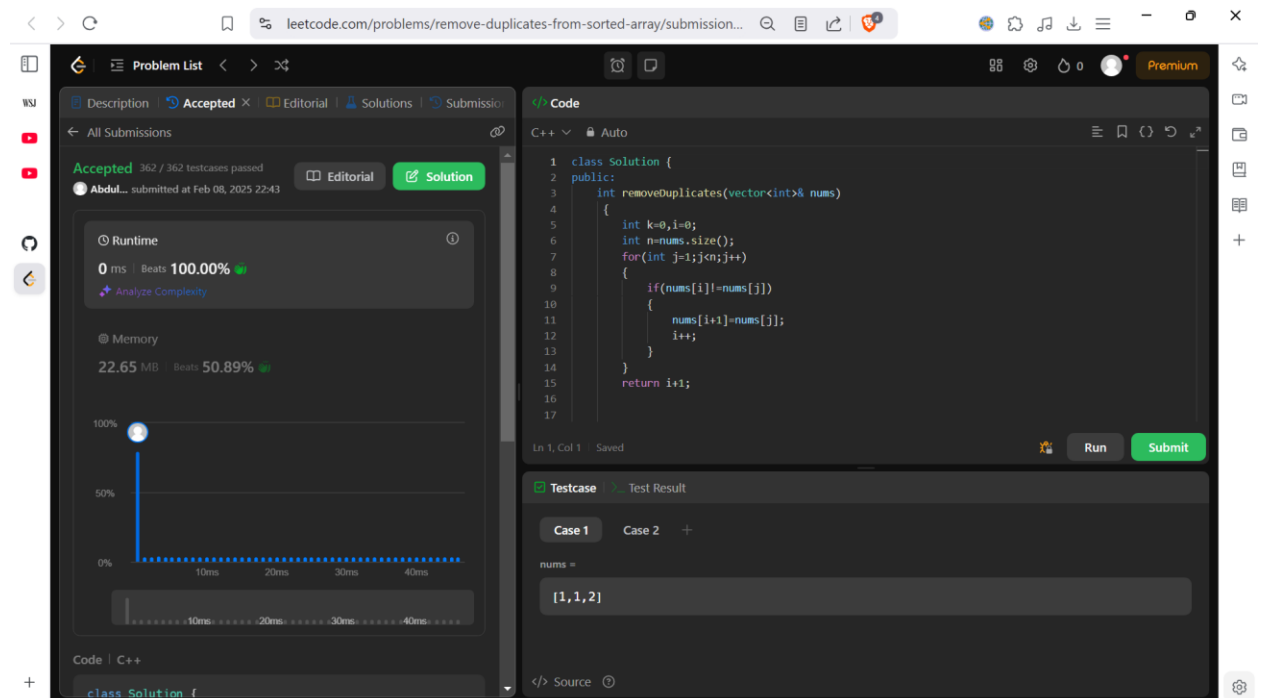
Date of Performance: 22/07/24

Subject Code: 22CSP – 351

DAY-1:

1. Remove Duplicates From The Sorted Array-

```
int removeDuplicates(vector<int>& nums) {  
    int k=0;int i=0;  
    int size=nums.size();  
    for(int j=1;j<size;j++){  
        if(nums[i]!=nums[j]){  
            nums[i+1]=nums[j];  
            i++;  
        }  
    }  
    return i+1;  
}
```



The screenshot shows a LeetCode submission page for the problem "Remove Duplicates from Sorted Array". The submission is accepted, with a runtime of 0 ms and a memory usage of 22.65 MB. The code is written in C++ and implements the two-pointer approach. The test case shows the input array [1, 1, 2] and the expected output is 3.

Problem List < > <>

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 362 / 362 testcases passed

Abdul... submitted at Feb 08, 2025 22:43

Editorial Solution

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

22.65 MB | Beats 50.89%

100% 50% 0%

10ms 20ms 30ms 40ms

Code | C++

```
class Solution {  
public:  
    int removeDuplicates(vector<int>& nums)  
    {  
        int k=0,i=0;  
        int n=nums.size();  
        for(int j=1;j<n;j++){  
            if(nums[i]!=nums[j])  
            {  
                nums[i+1]=nums[j];  
                i++;  
            }  
        }  
        return i+1;  
    }  
};
```

Ln 1, Col 1 Saved Run Submit

Testcase Test Result

Case 1 Case 2 +

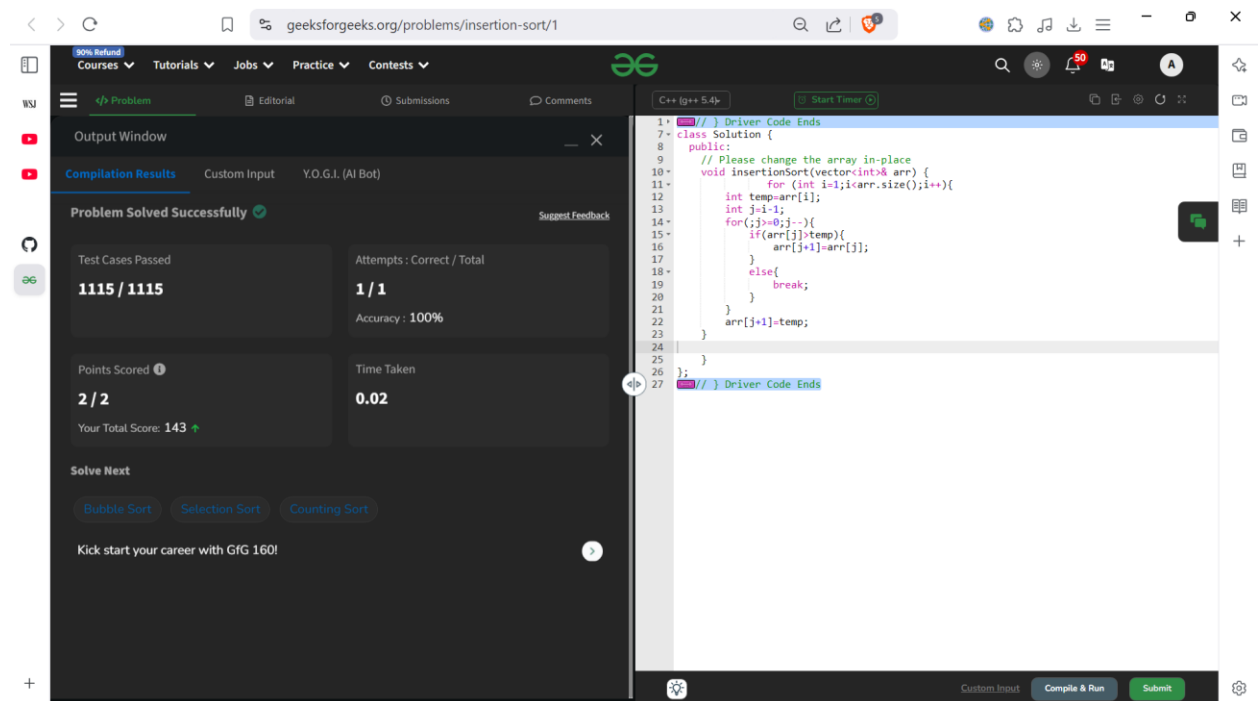
nums =

[1, 1, 2]

</> Source

2. Implementing Insertion Sort-

```
void insertionSort(vector<int>& arr) {
    // code here
    for (int i=1;i<arr.size();i++){
        int temp=arr[i];
        int j=i-1;
        for(;j>=0;j--){
            if(arr[j]>temp){
                arr[j+1]=arr[j];
            }
            else{
                break;
            }
        }
        arr[j+1]=temp;
    }
}
```



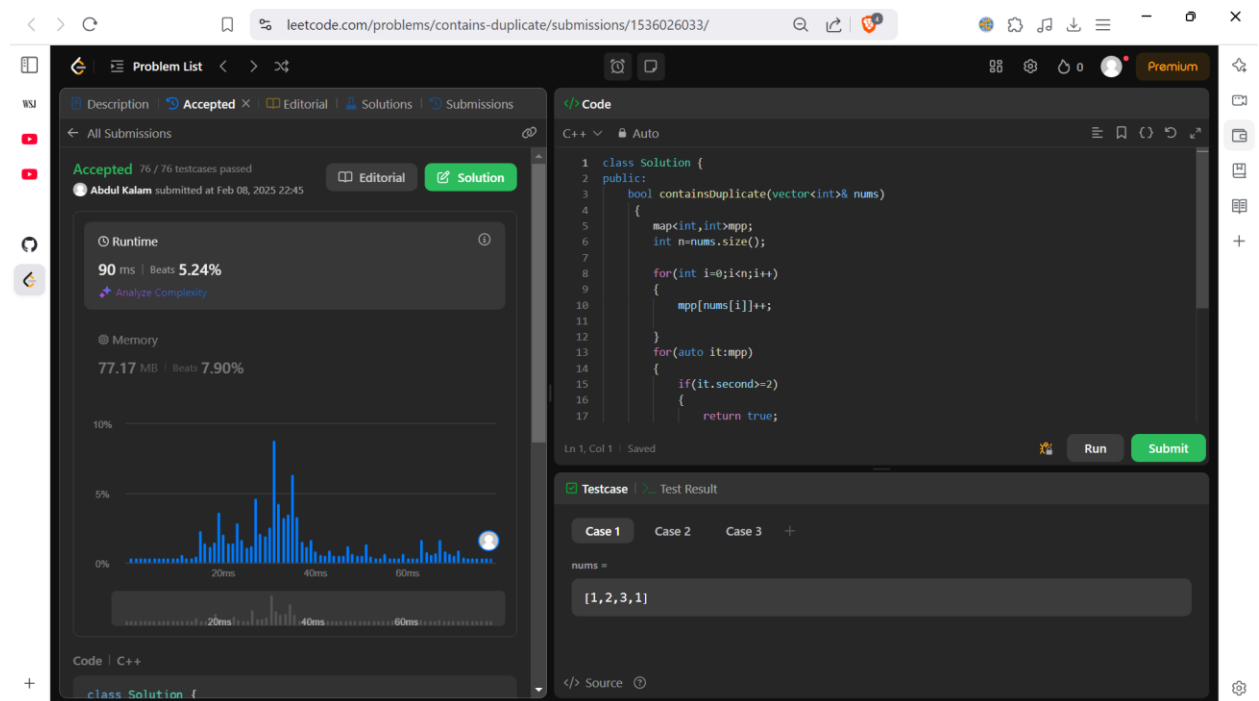
The screenshot shows a web-based IDE for C++ on the GeeksforGeeks website. The browser address bar shows the URL: <https://www.geeksforgeeks.org/problems/insertion-sort/1>. The IDE interface includes a top navigation bar with links for Courses, Tutorials, Jobs, Practice, and Contests. The main editor area displays the following C++ code:

```
1 // Driver Code Ends
2 class Solution {
3 public:
4     // Please change the array in-place
5     void insertionSort(vector<int>& arr) {
6         for (int i=1;i<arr.size();i++){
7             int temp=arr[i];
8             int j=i-1;
9             for(;j>=0;j--){
10                 if(arr[j]>temp){
11                     arr[j+1]=arr[j];
12                 }
13                 else{
14                     break;
15                 }
16             }
17             arr[j+1]=temp;
18         }
19     }
20 };
21 // Driver Code Ends
```

On the left side of the IDE, there is a sidebar showing the problem status: "Problem Solved Successfully". It also displays statistics: "Test Cases Passed: 1115 / 1115", "Attempts: Correct / Total: 1 / 1", "Accuracy: 100%", "Points Scored: 2 / 2", and "Time Taken: 0.02". At the bottom of the sidebar, there are links for "Solve Next" with options like "Bubble Sort", "Selection Sort", and "Counting Sort".

3. Contains Duplicate-

```
bool containsDuplicate(vector<int>& nums) {  
    map<int,int>mpp;  
    int n=nums.size();  
    for(int i=0;i<n;i++)  
    {  
        mpp[nums[i]]++;  
    }  
    for(auto it:mpp)  
    {  
        if(it.second>=2)  
        {  
            return true;  
        }  
    }  
    return false;  
}
```



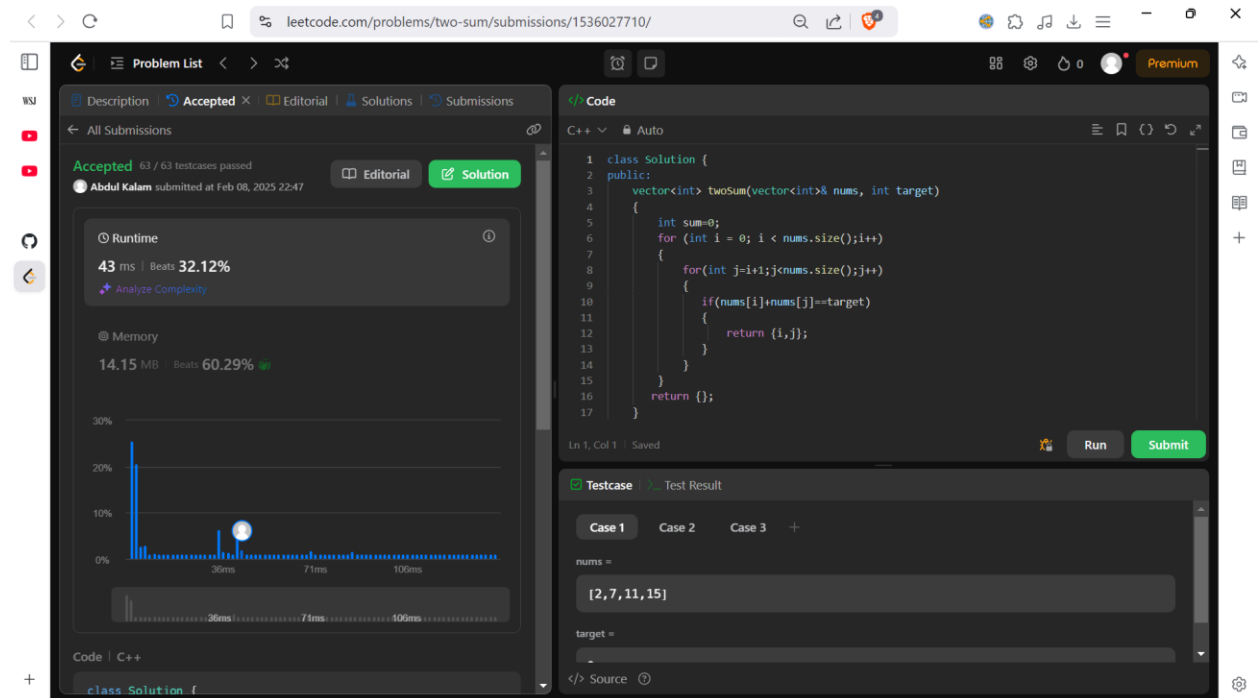
The screenshot displays the LeetCode interface for the 'Contains Duplicate' problem. The code editor shows the following C++ solution:

```
1 class Solution {  
2 public:  
3     bool containsDuplicate(vector<int>& nums)  
4     {  
5         map<int,int>mpp;  
6         int n=nums.size();  
7         for(int i=0;i<n;i++)  
8         {  
9             mpp[nums[i]]++;  
10        }  
11        for(auto it:mpp)  
12        {  
13            if(it.second>=2)  
14            {  
15                return true;  
16            }  
17        }  
18        return false;  
19    }  
20 }
```

The submission is accepted, with a runtime of 90 ms and memory usage of 77.17 MB. A test case [1, 2, 3, 1] is shown with the expected output true.

4. Two Sum –

```
vector<int> twoSum(vector<int>& nums, int target) {  
    int sum=0;  
    for (int i = 0; i < nums.size();i++)  
    {  
        for(int j=i+1;j<nums.size();j++)  
        {  
            if(nums[i]+nums[j]==target)  
            {  
                return {i,j};  
            }  
        }  
    }  
    return {};  
}
```



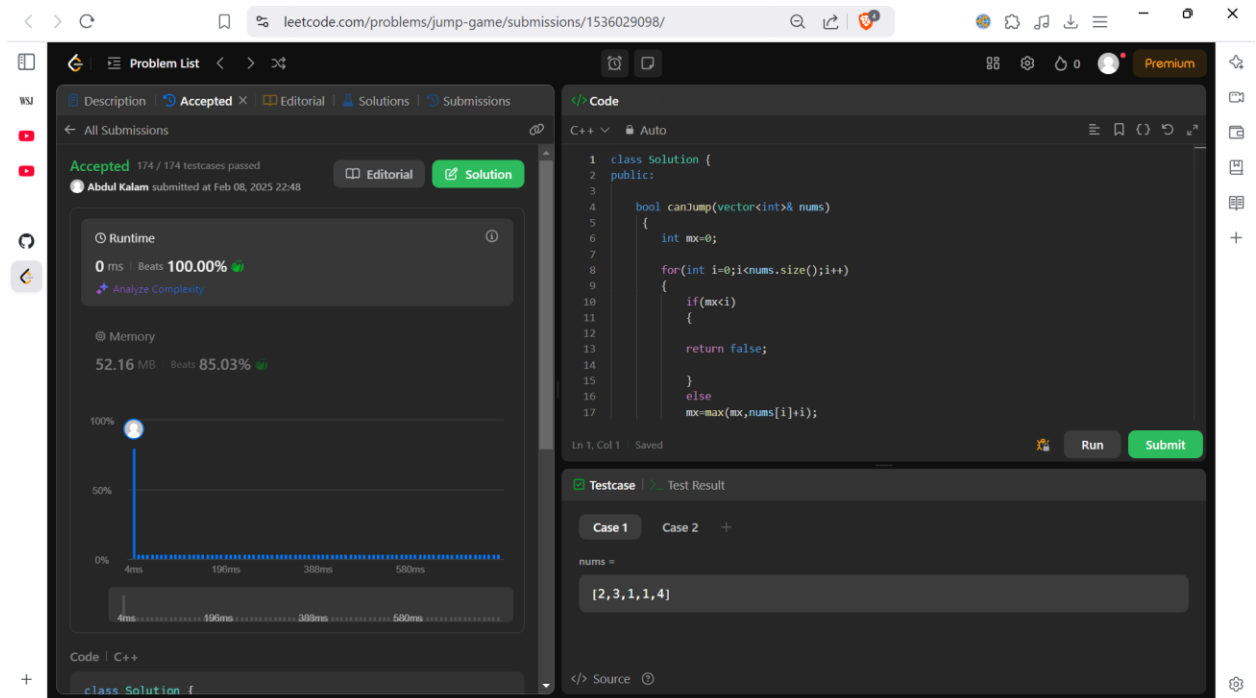
The screenshot shows a LeetCode submission for the "Two Sum" problem. The browser address bar displays the URL: `leetcode.com/problems/two-sum/submissions/1536027710/`. The submission is marked as "Accepted" and shows performance metrics: 43 ms runtime (beating 32.12%) and 14.15 MB memory (beating 60.29%). A runtime histogram is visible below the metrics. The code editor on the right contains the following C++ code:

```
1 class Solution {  
2 public:  
3     vector<int> twoSum(vector<int>& nums, int target)  
4     {  
5         int sum=0;  
6         for (int i = 0; i < nums.size();i++)  
7         {  
8             for(int j=i+1;j<nums.size();j++)  
9             {  
10                if(nums[i]+nums[j]==target)  
11                {  
12                    return {i,j};  
13                }  
14            }  
15        }  
16        return {};  
17    }  
18 }
```

Below the code editor, the "Testcase" section shows the input for Case 1: `nums = [2,7,11,15]` and `target = 9`. The "Run" and "Submit" buttons are visible at the bottom right of the code editor.

5. Jump Game-

```
bool canJump(vector<int>& nums) {  
    int mx=0;  
    for(int i=0;i<nums.size();i++)  
    {  
        if(mx<i)  
        {  
            return false;  
        }  
        else  
            mx=max(mx,nums[i]+i);  
    }  
    return true;  
}
```



The screenshot displays the LeetCode interface for the 'Jump Game' problem. The top navigation bar shows the problem name and submission link. The left sidebar contains tabs for 'Description', 'Accepted', 'Editorial', 'Solutions', and 'Submissions'. The main content area is divided into three sections: 'Runtime', 'Memory', and 'Testcase'.

Runtime: 0 ms | Beats 100.00% (Analyze Complexity)

Memory: 52.16 MB | Beats 85.03%

Testcase: Case 1: nums = [2, 3, 1, 1, 4]

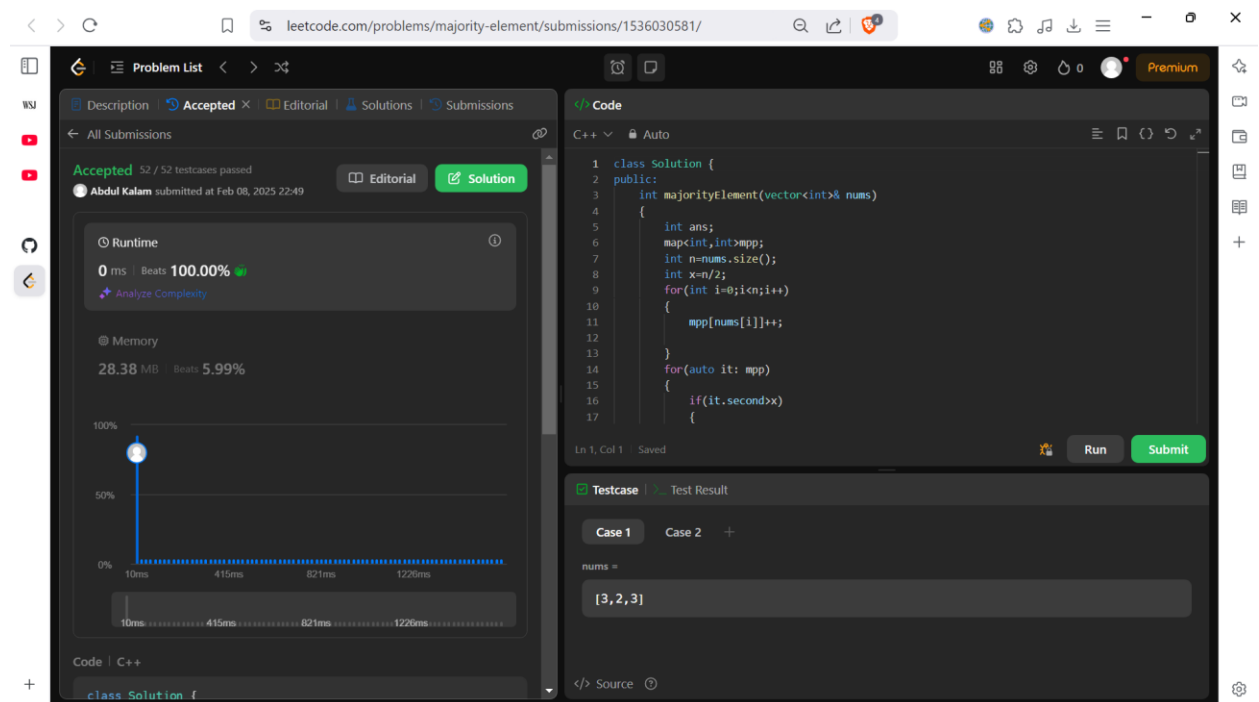
The C++ code editor shows the following solution:

```
1 class Solution {  
2 public:  
3     bool canJump(vector<int>& nums)  
4     {  
5         int mx=0;  
6         for(int i=0;i<nums.size();i++)  
7         {  
8             if(mx<i)  
9             {  
10                return false;  
11            }  
12            else  
13                mx=max(mx,nums[i]+i);  
14        }  
15    }  
16 }  
17
```

6. Majority Element-

```
int majorityElement(vector<int>& nums) {
    int ans;

    map<int,int>mpp;
    int n=nums.size();
    int x=n/2;
    for(int i=0;i<n;i++)
    {
        mpp[nums[i]]++;
    }
    for(auto it: mpp)
    {
        if(it.second>x)
        {
            ans=it.first;
        }
    }
    return ans;
}
```



leetcode.com/problems/majority-element/submissions/1536030581/

Problem List < > < > < >

Description Accepted x Editorial Solutions Submissions

All Submissions

Accepted 52 / 52 testcases passed
Abdul Kalam submitted at Feb 08, 2025 22:49

Editorial Solution

Runtime
0 ms | Beats 100.00%

Analyze Complexity

Memory
28.38 MB | Beats 5.99%

100%
50%
0%
10ms 415ms 821ms 1226ms

Code | C++

```
class Solution {
public:
    int majorityElement(vector<int>& nums)
    {
        int ans;
        map<int,int>mpp;
        int n=nums.size();
        int x=n/2;
        for(int i=0;i<n;i++)
        {
            mpp[nums[i]]++;
        }
        for(auto it: mpp)
        {
            if(it.second>x)
            {
                ans=it.first;
            }
        }
        return ans;
    }
};
```

Ln 1, Col 1 Saved Run Submit

Testcase Test Result

Case 1 Case 2 +

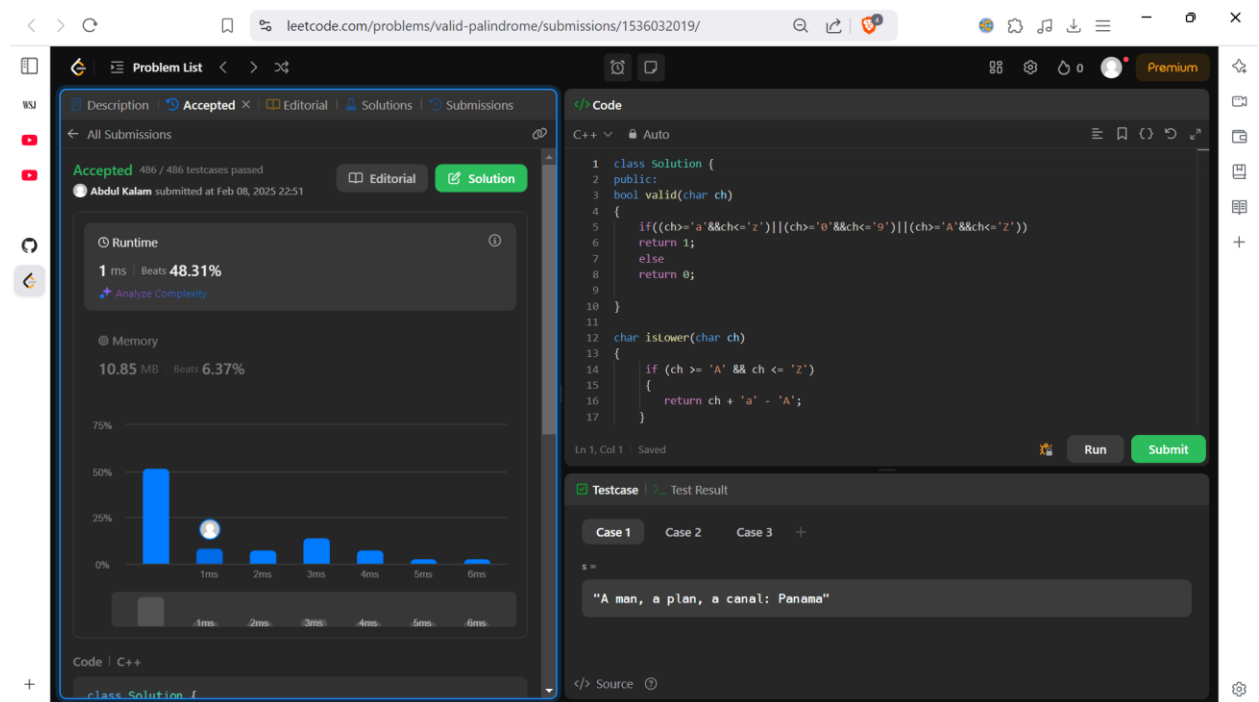
nums =

[3, 2, 3]

</> Source

7. Valid Palindrome –

```
class Solution {
public:
    bool valid(char ch) {
        return (ch >= 'a' && ch <= 'z') || (ch >= '0' && ch <= '9') || (ch >= 'A' && ch <= 'Z');
    }
    char isLower(char ch) {
        return (ch >= 'A' && ch <= 'Z') ? (ch + 'a' - 'A') : ch;
    }
    bool Palin(string str) {
        int s = 0, e = str.length() - 1;
        while (s < e) {
            if (str[s++] != str[e--]) return false;
            return true;
        }
        bool isPalindrome(string s) {
            string temp = "";
            for (char ch : s) {
                if (valid(ch)) temp.push_back(isLower(ch));
            }
            return Palin(temp);
        }
    };
};
```



leetcode.com/problems/valid-palindrome/submissions/1536032019/

Accepted 486 / 486 testcases passed
Abdul Kalam submitted at Feb 08, 2025 22:51

Runtime: 1 ms | Beats 48.31%

Memory: 10.85 MB | Beats 6.37%

```
class Solution {
public:
    bool valid(char ch) {
        return ((ch>='a'&&ch<='z')||((ch>='0'&&ch<='9'))||((ch>='A'&&ch<='Z')));
    }
    char isLower(char ch) {
        if (ch >= 'A' && ch <= 'Z')
            return ch + 'a' - 'A';
        return ch;
    }
    bool Palin(string str) {
        int s = 0, e = str.length() - 1;
        while (s < e) {
            if (str[s++] != str[e--]) return false;
            return true;
        }
        return true;
    }
};
```

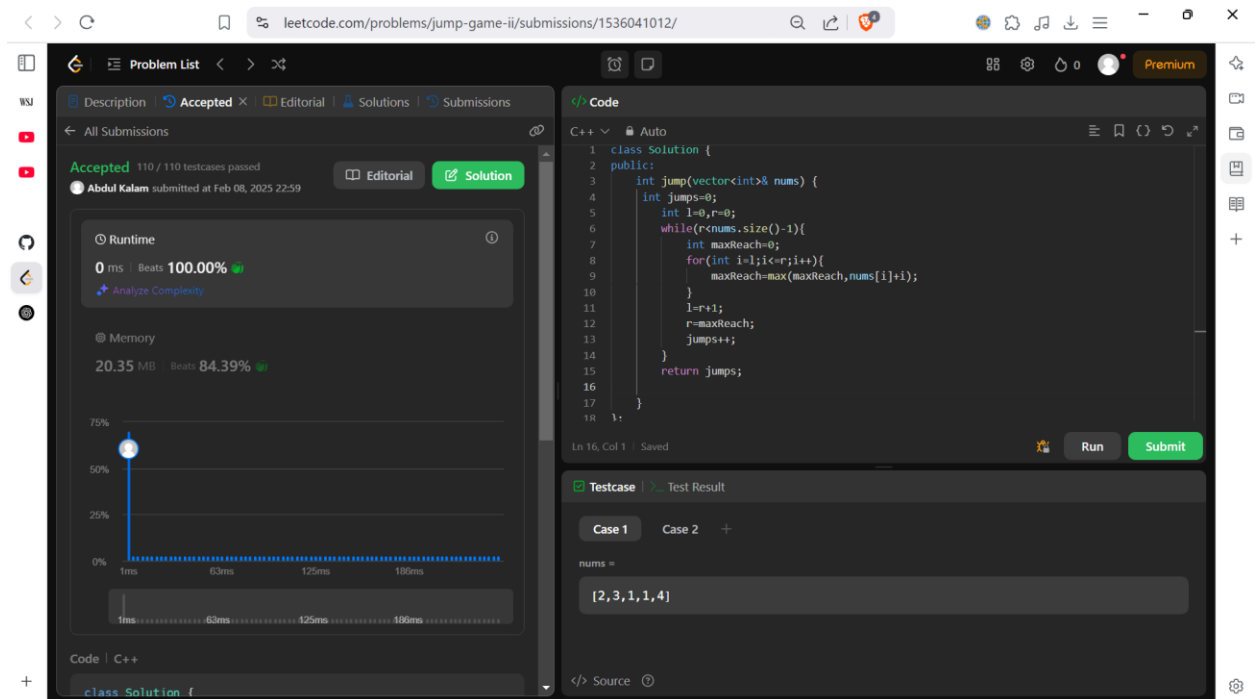
Testcase: Case 1 Case 2 Case 3 +

Input: "A man, a plan, a canal: Panama"

Output: Accepted

8. Jump Game 2-

```
int jump(vector<int>& nums) {  
    int jumps=0;  
    int l=0,r=0;  
    while(r<nums.size()-1){  
        int maxReach=0;  
        for(int i=l;i<=r;i++){  
            maxReach=max(maxReach,nums[i]+i);  
        }  
        l=r+1;  
        r=maxReach;  
        jumps++;  
    }  
    return jumps;  
}
```



The screenshot shows a LeetCode submission for the problem "Jump Game 2". The submission is accepted, with a runtime of 0 ms (beats 100.00%) and a memory usage of 20.35 MB (beats 84.39%). The code is written in C++ and implements the greedy algorithm for finding the minimum number of jumps to reach the end of the array. The test case shown is [2, 3, 1, 1, 4], which requires 2 jumps.

Runtime: 0 ms | Beats 100.00%
Memory: 20.35 MB | Beats 84.39%

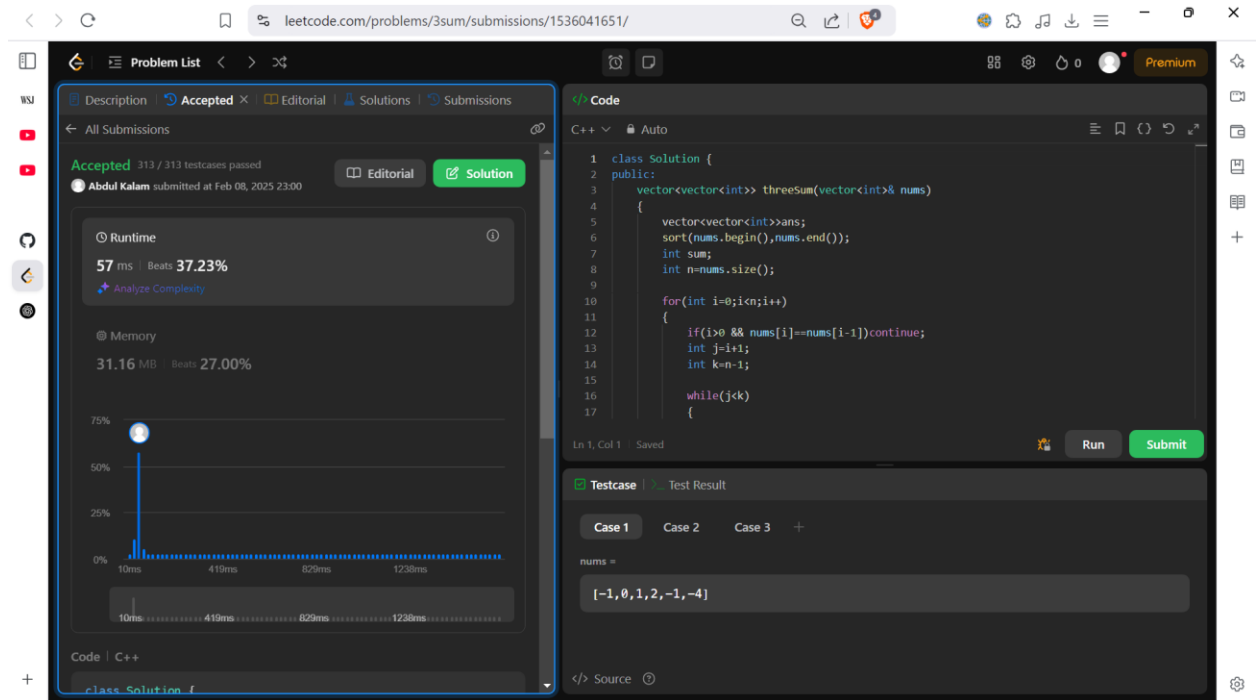
Code:

```
1 class Solution {  
2 public:  
3     int jump(vector<int>& nums) {  
4         int jumps=0;  
5         int l=0,r=0;  
6         while(r<nums.size()-1){  
7             int maxReach=0;  
8             for(int i=l;i<=r;i++){  
9                 maxReach=max(maxReach,nums[i]+i);  
10            }  
11            l=r+1;  
12            r=maxReach;  
13            jumps++;  
14        }  
15        return jumps;  
16    }  
17 }
```

Testcase: Case 1 Case 2 +
nums = [2, 3, 1, 1, 4]

9. 3 Sum-

```
vector<vector<int>> threeSum(vector<int>& nums) {  
    vector<vector<int>>ans;  
    sort(nums.begin(),nums.end());  
    int sum;  
    int n=nums.size();  
    for(int i=0;i<n;i++)  
    {  
        if(i>0 && nums[i]==nums[i-1])continue;  
        int j=i+1;  
        int k=n-1;  
        while(j<k)  
        {  
            sum=nums[i]+nums[j]+nums[k];  
            if(sum<0)  
            {  
                j++;  
            }  
            else if(sum>0)  
            {  
                k--;  
            }  
            else  
            {  
                vector<int> temp={nums[i],nums[j],nums[k]};  
                ans.push_back(temp);  
                j++;  
                k--;  
                while(j<k && nums[j]==nums[j-1])j++;  
                while(j<k && nums[k]==nums[k+1])k--;  
            }  
        }  
    }  
    return ans;  
}
```



10. Set Matrix Zeros-

```
void setZeroes(vector<vector<int>>& matrix) {
```

```
    int row=matrix.size();
```

```
    int col=matrix[0].size();
```

```
    vector<int> indexRow(row,0);
```

```
    vector<int> indexCol(col,0);
```

```
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(matrix[i][j]==0){
                indexRow[i] =1;
                indexCol[j]=1;
            }
        }
    }
}
```

```
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        if(indexRow[i] || indexCol[j]){  
            matrix[i][j]=0;  
        }  
    }  
}
```

The screenshot displays the LeetCode submission page for the problem "Set Matrix Zeroes". The submission is by user "Abdul Kalam" and is marked as "Accepted". The runtime is 0 ms, beating 100.00% of submissions, and the memory usage is 18.80 MB, beating 25.13% of submissions. The test case shows a 3x3 matrix: `[[1,1,1],[1,0,1],[1,1,1]]`.

Code:

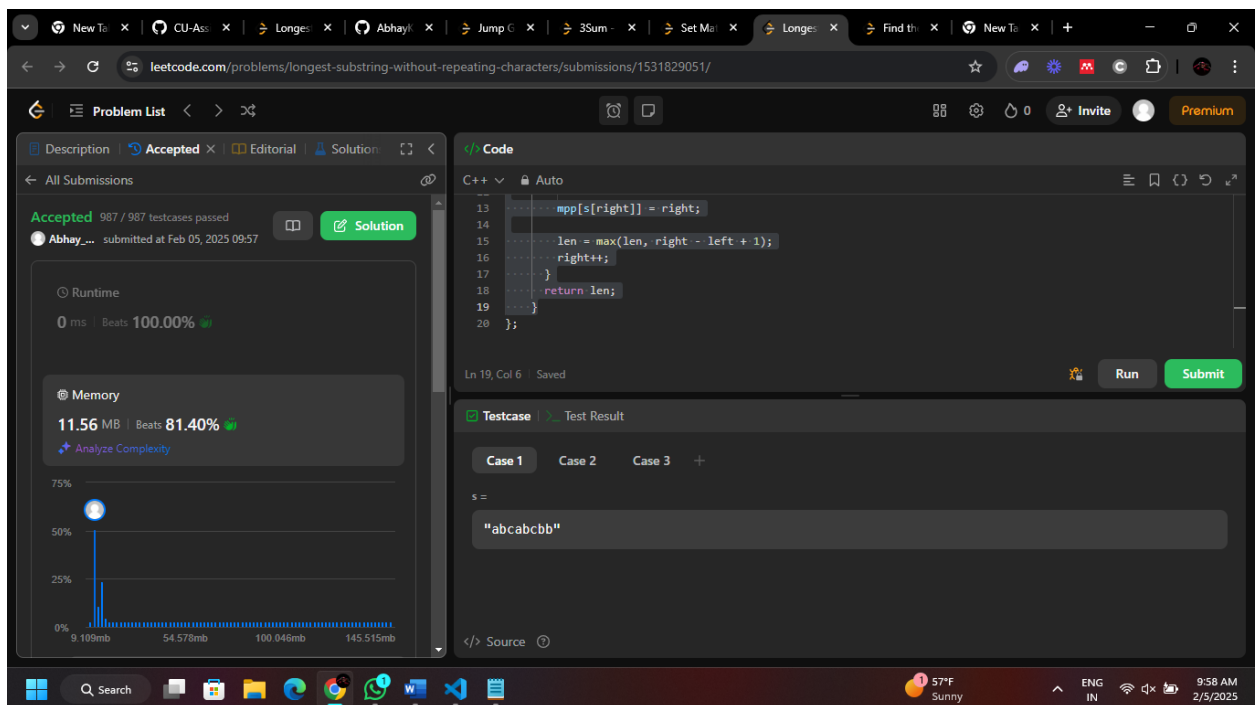
```
C++  
class Solution {  
public:  
    void setZeroes(vector<vector<int>>& matrix) {  
        int indexRow[matrix.size()] = {0};  
        int indexCol[matrix[0].size()] = {0};  
        for(int i=0; i<matrix.size(); i++){  
            for(int j=0; j<matrix[0].size(); j++){  
                if(indexRow[i] || indexCol[j]){  
                    matrix[i][j]=0;  
                }  
            }  
        }  
    }  
};
```

Testcase:

Case 1: `matrix = [[1,1,1],[1,0,1],[1,1,1]]`

11. Longest Substring Without Repeating Characters-

```
int lengthOfLongestSubstring(string s) {  
    vector<int> mpp(256, -1);  
  
    int left = 0, right = 0;  
    int n = s.size();  
    int len = 0;  
    while (right < n) {  
        if (mpp[s[right]] != -1)  
            left = max(mpp[s[right]] + 1, left);  
  
        mpp[s[right]] = right;  
  
        len = max(len, right - left + 1);  
        right++;  
    }  
    return len;  
}
```



The screenshot displays a web browser window with multiple tabs open. The active tab is the LeetCode submission page for the problem "Longest Substring Without Repeating Characters". The submission status is "Accepted", indicating that all 987 test cases were passed. The user's name, "Abhay...", is visible next to the submission time, "submitted at Feb 05, 2025 09:57".

The code editor shows the following C++ code:

```
13 mpp[s[right]] = right;  
14  
15 len = max(len, right - left + 1);  
16 right++;  
17 }  
18 return len;  
19 }  
20 }
```

The runtime and memory usage are displayed on the left side of the page. The runtime is 0 ms, which beats 100.00% of other submissions. The memory usage is 11.56 MB, which beats 81.40% of other submissions. A graph below the memory usage shows the memory consumption over time, with a peak of 145.515mb.

The test case section shows the input string "s = 'abcabcbb'". The "Run" and "Submit" buttons are visible at the bottom right of the code editor.

12. Finding Duplicate Number-

```
int findDuplicate(vector<int>& nums) {
    int ans;
    map<int,int>mpp;
    int n=nums.size();
    for(int i=0;i<n;i++)
    {
        mpp[nums[i]]++;
    }
    for(auto it : mpp)
    {
        if(it.second>=2)
        {
            ans=it.first;
        }
    }
    return ans;
}
```

Screenshot of the LeetCode interface showing the solution for the problem "Find the Duplicate Number" (1536046450).

The interface is divided into two main sections: the left sidebar and the main content area.

Left Sidebar (Problem List):

- Problem List: Accepted, Editorial, Solutions, Submissions
- All Submissions
- Accepted (59 / 59 testcases passed)
- Abdul Kalam submitted at Feb 08, 2025 23:04
- Editorial Solution
- Runtime: 268 ms | Beats 5.03% (Analyze Complexity)
- Memory: 107.19 MB | Beats 5.59%
- Graph showing runtime distribution (0% to 40%)
- Code | C++

Main Content Area (Code Editor):

```
13 for(auto it : mpp)
14 {
15     if(it.second>=2)
16     {
17         ans=it.first;
18     }
19 }
20
21
22 }
23 return ans;
24 }
25 ;
```

Ln 23, Col 16 | Saved

Run Submit

Testcase | Test Result

Case 1 Case 2 Case 3 +

nums =

[1,3,4,2,2]

</> Source

Problem List

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 168 / 168 testcases passed

Abdul Kalam submitted at Feb 08, 2025 23:06

Editorial Solution

Runtime

1 ms Beats 10.09%

Analyze Complexity

Memory

17.02 MB Beats 11.33%

Runtime	Beats
1ms	10.09%
2ms	
3ms	
4ms	

Code | C++

Code

C++ v Auto

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode() : val(0), next(nullptr) {}
7  *     ListNode(int x) : val(x), next(nullptr) {}
8  *     ListNode(int x, ListNode *next) : val(x), next(next) {}
9  * };
10 */
11 class Solution {
12 public:
13     ListNode* deleteDuplicates(ListNode* head)
14     {
15         set<int> s;
16         ListNode* temp=head;
17     }
18 }
```

Ln 1, Col 1 Saved Run Submit

Testcase | Test Result

Case 1 Case 2 +

head =

[1,1,2]

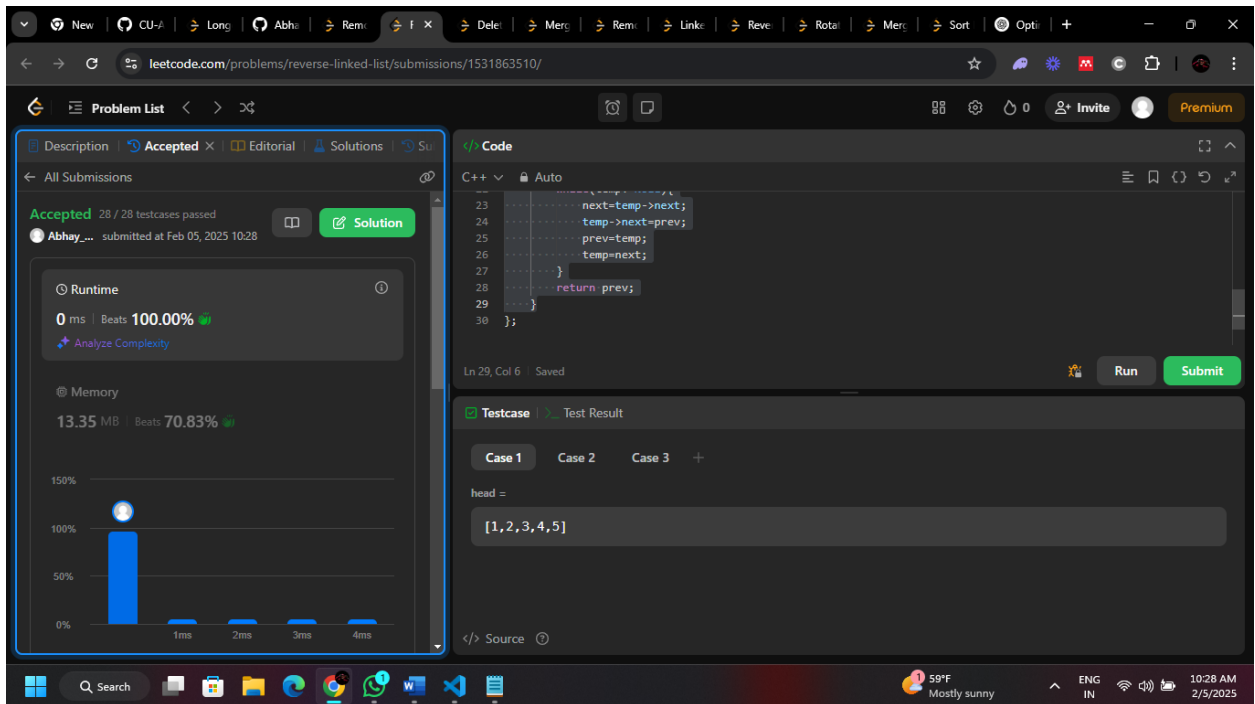
</> Source

2. Reverse A Linked List –

```
ListNode* reverseList(ListNode* head) {  
    if(head==NULL || head->next==NULL){  
        return head;  
    }  
}
```

```
ListNode* prev=NULL;  
ListNode* temp=head;  
ListNode* next=NULL;
```

```
while(temp!=NULL){  
    next=temp->next;  
    temp->next=prev;  
    prev=temp;  
    temp=next;  
}  
return prev;  
}
```



The screenshot displays a web browser window with the URL <https://leetcode.com/problems/reverse-linked-list/submissions/1531863510/>. The page shows a C++ solution for the "Reverse Linked List" problem. The solution is accepted, with a runtime of 0 ms and memory usage of 13.35 MB. The code is shown in the editor, and the test case [1, 2, 3, 4, 5] is displayed.

Accepted 28 / 28 testcases passed
Abhey... submitted at Feb 05, 2025 10:28

Runtime
0 ms | Beats 100.00%

Memory
13.35 MB | Beats 70.83%

Code

```
23 ...->next=temp->next;  
24 ...->next=prev;  
25 ...->next=temp;  
26 ...->next=temp->next;  
27 ...}  
28 ...return prev;  
29 ...}  
30 ...;
```

Testcase | Test Result

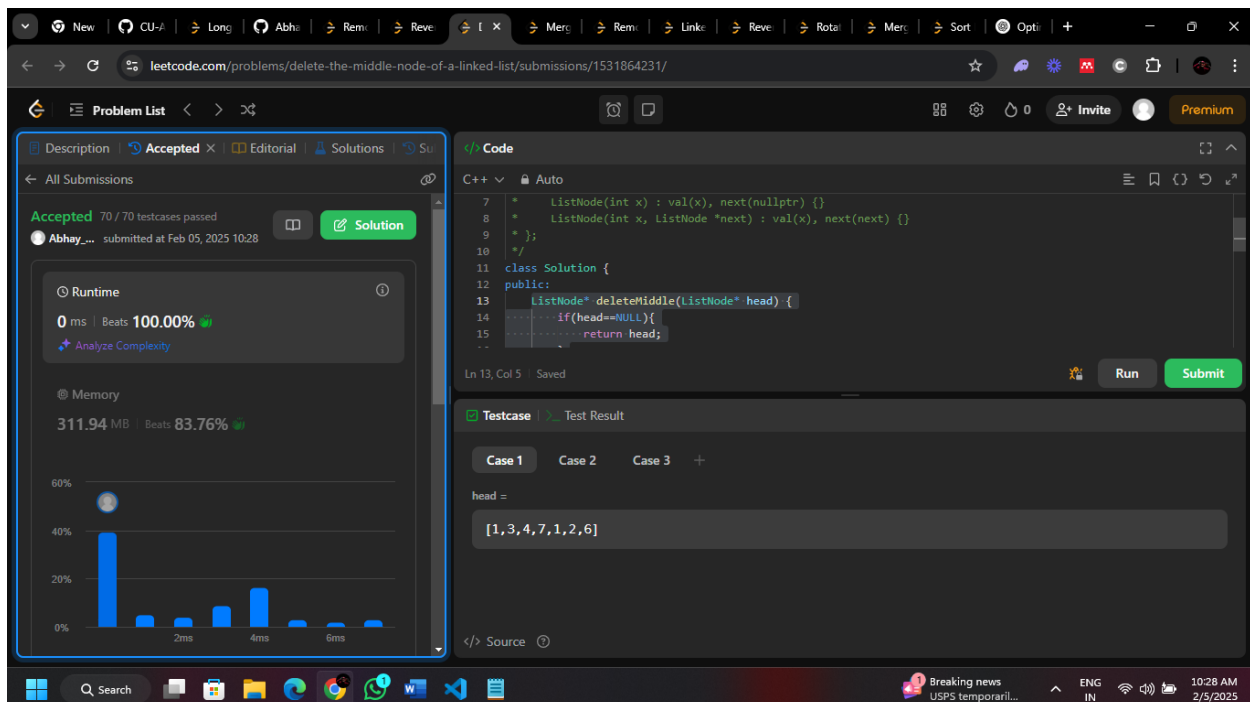
Case 1 Case 2 Case 3 +

head =

[1, 2, 3, 4, 5]

3. Delete Middle Node Of A List-

```
ListNode* deleteMiddle(ListNode* head) {
    if(head==NULL){
        return head;
    }
    if(head->next==NULL){
        head=head->next;
        return head;
    }
    ListNode* fast=head;
    ListNode* slow=head;
    ListNode* prev=NULL;
    while(fast!=NULL && fast->next!=NULL){
        fast=fast->next->next;
        prev=slow;
        slow=slow->next;
    }
    prev->next=slow->next;
    slow=slow->next;
    return head; }
```



The screenshot displays a LeetCode submission for the problem "Delete the Middle Node of a Linked List". The submission is marked as "Accepted" with 70/70 test cases passed. The runtime is 0 ms, which is 100.00% faster than other submissions. The memory usage is 311.94 MB, which is 83.76% less than other submissions. The code is written in C++ and implements the deleteMiddle function. The test case shows a linked list [1, 3, 4, 7, 1, 2, 6] and the expected output is [1, 3, 4, 7, 1, 2, 6].

4. Merge Two Sorted Linked List-

```
ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {  
    if (list1 == NULL && list2 == NULL) {  
        return NULL;  
    }  
  
    if (list1 == NULL) {  
        return list2;  
    }  
    if (list2 == NULL) {  
        return list1;  
    }  
  
    ListNode* dummy = new ListNode(-1);  
    ListNode* head = dummy;  
  
    while (list1 != NULL && list2 != NULL) {  
        if (list1->val <= list2->val) {  
            head->next = list1;  
            list1 = list1->next;  
        } else {  
            head->next = list2;  
            list2 = list2->next;  
        }  
        head = head->next;  
    }  
  
    if (list1 != NULL) {  
        head->next = list1;  
    } else if (list2 != NULL) {  
        head->next = list2;  
    }  
  
    return dummy->next;  
}
```



Discover. Learn. Empower.

Screenshot of a web browser displaying the LeetCode interface for the "Merge Two Sorted Lists" problem. The browser's address bar shows the URL: `leetcode.com/problems/merge-two-sorted-lists/submissions/1531865287/`. The page is divided into several sections:

- Problem List:** Located at the top, it includes tabs for "Description", "Accepted" (selected), "Editorial", and "Solution". Below these tabs is a link to "All Submissions".
- Accepted Submissions:** A section showing the submission status "Accepted" for 208 out of 208 testcases. The submission is by user "Abhay_..." and was submitted on Feb 05, 2025 at 10:29. A green "Solution" button is visible.
- Runtime and Memory:** The "Runtime" section shows a time of 0 ms and a success rate of 100.00%. The "Memory" section shows a memory usage of 19.48 MB and a success rate of 62.67%.
- Code Editor:** The main area displays the C++ code for the solution. The code uses a dummy node and iterates through the two input lists, merging them into a single sorted list. The code is as follows:

```
C++
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        if (list1 == NULL) return list2;
        if (list2 == NULL) return list1;
        if (list1->val < list2->val) {
            list1->next = mergeTwoLists(list1->next, list2);
        } else if (list2->val < list1->val) {
            list2->next = mergeTwoLists(list1, list2->next);
        } else {
            list1->next = list2;
        }
        return list1;
    }
};
```
- Testcase:** A section showing the test results for the submission. It includes a table with columns for "Case 1", "Case 2", and "Case 3". The test results show that the submission passed all three cases.

5. Detect A Cycle In A Linked List-

```
bool hasCycle(ListNode *head) {
```

```
    ListNode* fast=head;
```

```
    ListNode* slow=head;
```

```
    while(fast!=NULL && fast->next!=NULL){
```

```
        fast=fast->next->next;
```

```
        slow=slow->next;
```

```
        if(slow==fast){
```

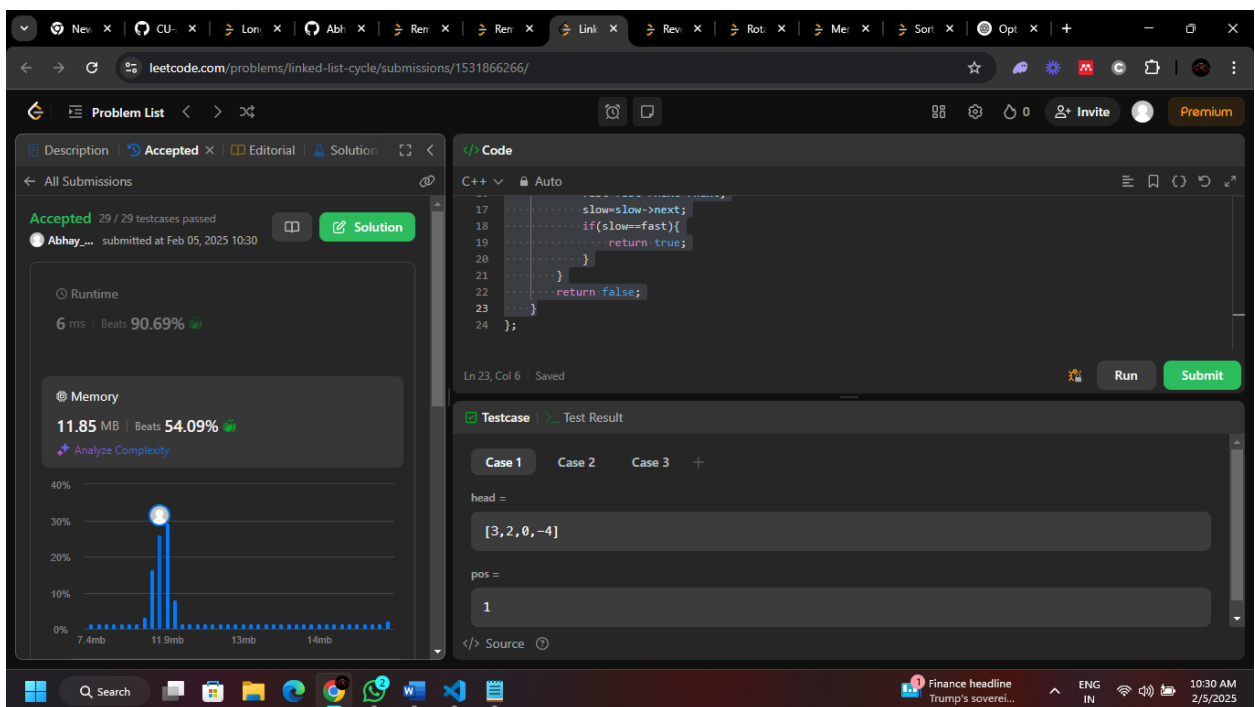
```
            return true;
```

```
        }
```

```
    }
```

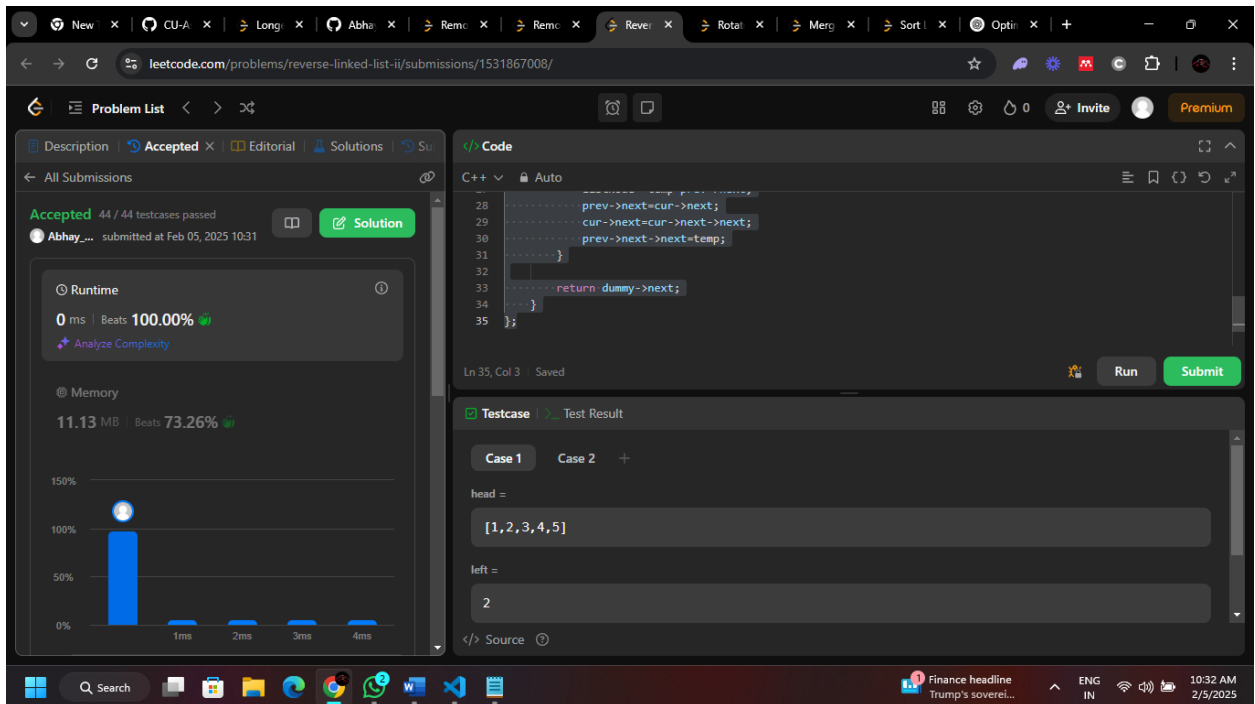
```
    return false;
```

```
}
```



6. Reverse Linked List 2-

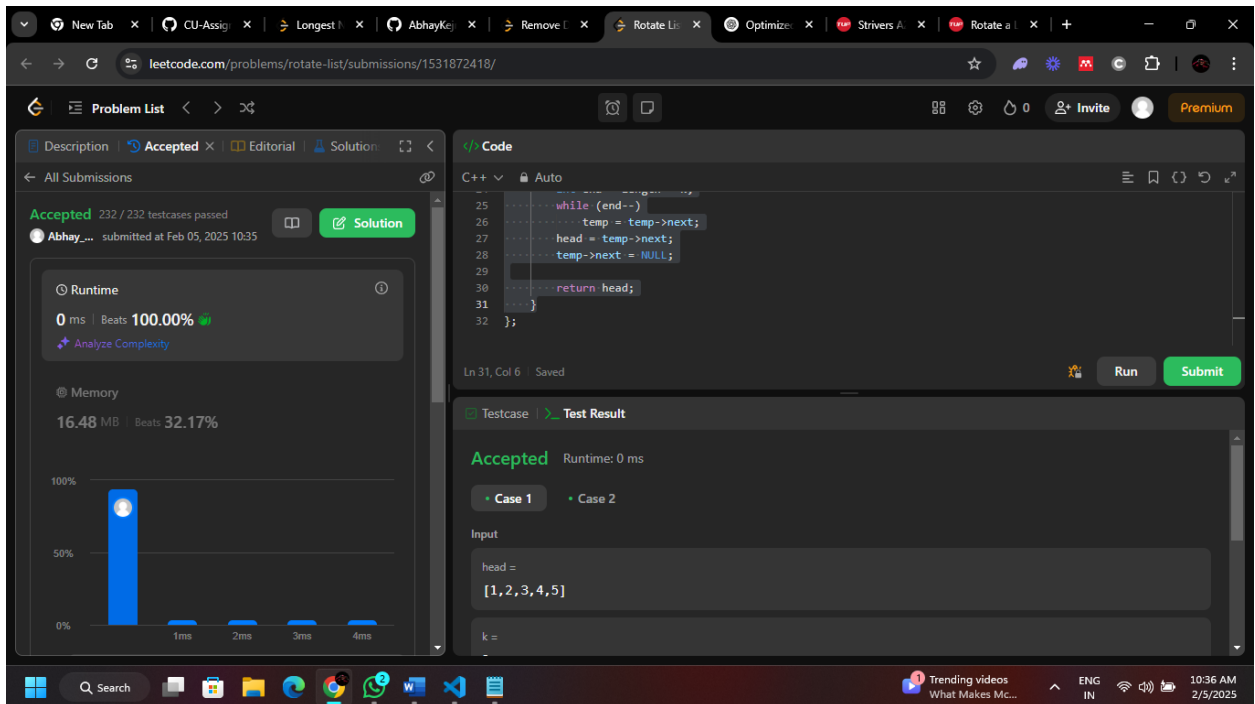
```
ListNode* reverseBetween(ListNode* head, int left, int right) {
    if(head==NULL || head->next==NULL){
        return head;
    }
    ListNode* dummy=new ListNode(-1);
    dummy->next=head;
    ListNode* prev=dummy;
    for(int i=1;i<left;i++){
        prev=prev->next;
    }
    ListNode* cur=prev->next;
    for(int i=0;i<right-left;i++){
        ListNode* temp=prev->next;
        prev->next=cur->next;
        cur->next=cur->next->next;
        prev->next->next=temp;
    }
    return dummy->next;
}
```



The screenshot displays a web browser window with multiple tabs. The active tab is a LeetCode submission page for the problem "Reverse Linked List II". The submission is marked as "Accepted" with 44/44 testcases passed. The user "Abhey..." submitted it on Feb 05, 2025 at 10:31. The runtime is 0 ms, beating 100.00% of solutions, and the memory usage is 11.13 MB, beating 73.26%. The code is written in C++ and implements the reverseBetween function. The test case shows head = [1,2,3,4,5] and left = 2, resulting in [1,3,2,4,5]. The browser's taskbar at the bottom shows various application icons and the system clock indicating 10:32 AM on 2/5/2025.

7. Rotate A List-

```
ListNode* rotateRight(ListNode* head, int k) {  
    if (head == NULL || head->next == NULL || k == 0)  
        return head;  
    ListNode* temp = head;  
    int length = 1;  
    while (temp->next != NULL) {  
        ++length;  
        temp = temp->next;  
    }  
    temp->next = head;  
    k = k % length;  
    int end = length - k;  
    while (end-->0) {  
        temp = temp->next;  
        head = temp->next;  
        temp->next = NULL;  
    }  
    return head;  
}
```



The screenshot shows a web browser with multiple tabs, including 'New Tab', 'CU-Assign', 'Longest', 'AbhayK', 'Remove', 'Rotate Li', 'Optimize', 'Strivers A', and 'Rotate a'. The active tab is 'Rotate Li' on the LeetCode website, displaying the 'rotate-list' problem submission page for user 'AbhayK'.

The submission is marked as 'Accepted' with 232 / 232 testcases passed. The runtime is 0 ms, and the memory usage is 16.48 MB. The code is written in C++ and implements the rotateRight function. The test case input is head = [1, 2, 3, 4, 5] and k = 2, resulting in an accepted output.

The code snippet shown is:

```
C++  
25 while (end-->0) {  
26     temp = temp->next;  
27     head = temp->next;  
28     temp->next = NULL;  
29 }  
30 return head;  
31 }  
32 }
```

The test case input is:

```
head =  
[1, 2, 3, 4, 5]  
k =  
2
```

The test case output is 'Accepted' with a runtime of 0 ms.

8. Sort List –

```
ListNode* findMiddle(ListNode* head){
    ListNode* slow=head;
    ListNode* fast=head->next;

    while(fast!=NULL && fast->next!=NULL){
        slow=slow->next;
        fast=fast->next->next;
    }
    return slow;
}

ListNode* mergeTwoList(ListNode* left, ListNode* right){
    ListNode* dummy=new ListNode(-1);
    ListNode* temp=dummy;

    while(left!=NULL && right!=NULL){
        if(left->val < right->val){
            temp->next=left;
            temp=left;
            left=left->next;
        }
        else{
            temp->next=right;
            temp=right;
            right=right->next;
        }
    }

    if(left)temp->next=left;
    else temp->next=right;

    return dummy->next;
}

ListNode* sortList(ListNode* head) {
    if (head==NULL || head->next==NULL)return head;
```

Windows taskbar showing search, file explorer, and various application icons. System tray includes language (ENG IN), network, volume, and date/time (10:32 AM 2/5/2025).